High-Precision End-to-End Latency Guarantees Using Packet Wash

Lijun Dong, Alexander Clemm
Futurewei Technologies Inc.
2330 Central Expressway
Santa Clara, CA, U.S.A
{lijun.dong, alex}@futurewei.com

***Abstract*— Many future networking applications demand high-precision end-to-end latency guarantees. Past QoS schemes have focused on scheduling and prioritization schemes for packets in their entirety but further advances are needed. Packet Wash is a recently proposed technology that allow for selective prioritization of payload chunks within packets themselves. Its primary intention is to reduce the number of packet drops when network congestion occurs by providing an option to drop less important payload portions first. However, Packet Wash opens up other possibilities. This paper introduces the use of Packet Wash to facilitate high-precision end-to-end latency guarantees. It involves a novel QoS algorithm that allows to reduce packet dwell time in routers by applying Packet Wash operations when the packet is in danger of being late, trading off transmission of low-priority chunks against improved latency.**

Keywords— New IP, BPP, high-precision networking, latency guarantees, dwell time, QoS, Qualitative Communications, packet wash

I. INTRODUCTION

Today's Internet technology is based on the best effort principle (BE), which attempts to transport packets to their intended destinations to the network's best ability without any assurance of the quality of service in such terms as end-to-end latency or throughput. There is no denying that this approach has been immensely successful. Not only has it proven highly resilient to perturbations such as link failures and fluctuations in traffic, but also it has allowed many services and applications to converge on the same networking infrastructure over the years. Underlying advances in link speeds and node reliability have even allowed interactive near-real time services such as voice and video conferencing to be supported, which was originally thought of as almost impossible to do.

However, at the same time Internet Technology is increasingly running into limitations that call into questions its ability to continue to support ever-increasing demands of future Internet applications. Many of those applications are characterized by extreme demands in terms of service levels that need to be supported by the network [1]. For example, Holographic-Type Communication (HTC) requires extremely low latency to be able to dynamically adjust content that is being streamed based on user interaction such as changes in viewing position and angle [2]. Operational Technology and Industry 4.0 applications, such as remote controllers for high-precision assembly lines or robotic arms, require very precise synchronization that today prevent them from being hosted in the cloud or even away from the factory floor [3]. The Tactile Internet includes applications to remotely control machinery using haptic feedback, which requires control loops with budgets of only very few milliseconds [4]. At the same time, many of those applications are also very sensitive to violations of service level objectives, resulting not merely in a degraded experience but leaving applications virtually unusable. An example concerns tactile Internet application, where missed latency objectives result in an abrupt loss of the user's sensation of haptic feedback required to operate remote machinery safely.

As a result, high-precision networking that allows to deliver communication services that are able to comply with stringent service objectives is receiving renewed attention. The ability to deliver such services is a prerequisite to unlock the economic potential of future networking applications but simply cannot be supported with current Internet Technology whose best-effort paradigm results in service levels that cannot be guaranteed with sufficient accuracy.

A major obstacle to high-precision networking in the Internet results from the unpredictable nature of packet collisions. When such collisions occur, packets must be queued in a certain order until they can be transmitted. The depth of the queue, i.e. the amount of data that is ahead of the packet, affects the packet's latency and is the dominating factor in the packet's dwell time, i.e. the amount of time that it spends in a node after its initial reception until it is forwarded. QoS techniques generally involve scheduling algorithms that determine which packet gets to enter a queue first, as well as prioritization schemes among multiple queues transmitting over the same interface. The path to high-precision networking lies in advances of such QoS algorithms.

Recently, a new technology referred to as Qualitative Communication has been introduced. It is based on the idea of allowing to structure packet payloads into chunks and associate those chunks with different priorities. This provides the option of selectively dropping a lower-priority chunk from a packet when congestion occurs, as opposed to having to drop a packet in its entirety. The operation to remove low-priority chunks from a packet is also referred to as "Packet Wash". Dropped chunks are not important enough to trigger packet retransmission, thus protecting low latency characteristics. At the same time, the remaining (high-priority) chunks will by themselves still be useful enough for the receiving application to continue to function properly, with lost chunks affecting the experienced quality only slightly. It is up to the sending application to

978-3-903176-32-4

determine how to chunkify payload, specifically, which chunks to label as less important (can be afforded to be dropped) than others (which would affect applications more severely if dropped and have to be retransmitted).

Packet wash was originally conceived as a way to reduce packet loss and offer a way to salvage essential portions of packets when collision occurs. However, it opens up additional possibilities. Applying packet wash operations to packets reduces the amount of data that needs to be transmitted, which in turn reduces the amount of time that it takes to transmit the packet and hence its dwell time. In addition to reducing its own dwell time, it also reduces the dwell time of every packet behind it in the queue.

While packet wash operations should not be applied indiscriminately, this does open the possibility to apply packet wash operations when critical packets are in danger of missing their latency objective. While the amount of time that can be shaved off may be insignificant for an individual packet, this time may add up when encountering queues of significant depth that contain multiple packets to which wash operations can be applied. This leads to a new QoS technique that facilitates the meeting of end-to-end latency objectives by applying wash operations to packets in a queue when required. We introduce this technique in this paper.

The remainder of this paper is structured as follows: Section II provides an overview of related work. Section III gives some further background on "Packet Wash" and its enabling technologies. Section IV presents in detail our algorithm and system design. As the QoS technique needs to be applied at line-rate, our system design takes into account the ability to be easily implemented in hardware. Section V provides an assessment of performance and effectiveness of our technique. Finally, Section VI provides an outlook for future work and conclusions.

II. Related Work

A lot of networking research over the past decades has centered around the topic of QoS, devising technology that helps to improve and optimize the quality with which best-effort networking services are delivered to applications and navigating tradeoffs such as utilization or fairness.

To this end, IETF has defined two complementary foundational QoS architectures, DiffServ [5] and IntServ. DiffServ is a multiplexing technique that is used to manage resources such as bandwidth and queueing buffers between different classes of traffic. DiffServ includes a feature, Explicit Congestion Notification (ECN) [6], that utilizes two bits in an IP packet's DiffServ field to indicate to an end node that congestion is being experienced, aiming to reduce packet loss and delay by driving the sender to lower its transmission rate until the congestion clears, without dropping packets. However, ECN does not mitigate situations in which a congestion has already occurred, which may still result in packet drops and the need for retransmission, as well as in reduced throughput when transmission rates get throttled too conservatively.

IntServ includes two services, Controlled Load Service [7] and Guaranteed Service (GS) [8]. The latter is of particular relevance, based on the concept of reserving resources in advance to provide per-flow fixed bandwidth guarantees. GS traffic is shaped at the ingress network edge, so the flow does not consume more resources than have been reserved. To support latency guarantees, flows need to be re-shaped on every hop as collisions and resource contention between packets could still occur and lead to the possibility of loss and unpredictable latency variations. GS does not provide specific support for dynamic bitrate adjustments and is arguably mainly suited for constant bit-rate (CBR) traffic. Furthermore, no consideration is given to specific end-to-end latency requirements of specific packets or flows.

More recently, the IETF DetNet Working Group has proposed the Deterministic Networking Architecture (DetNet) [9]. The DetNet architecture intends to provide per-flow service guarantees in terms of bounded delay, i.e. worst-case end-to-end latency, as well as packet loss ratio and bounds on out-of-order packet delivery. Similar to GS, DetNet's most fundamental limitation is in its targeted scope of CBR reservations, whereas many future applications involve highly variable bitrates.

The rise of Software-Defined Networking (SDN) has led to the introduction of alternative network architectures in which a centralized controller ("God Box") is able to provision paths in near-real time. The fact that the controller possesses full information about the entire network holds the promise that the controller will be able to provide global optimization for the way in which traffic is steered and provide precise control over what service levels are incurred. In contrast, similar results would be much harder to accomplish with a decentralized control plane in which nodes have only very limited and localized information. However, practical challenges that have not been adequately addressed include the need to scale to millions of simultaneous flows and inability to adapt to conditions in real time. Scheduling and forwarding decisions need to be made at line rate and in sub-microsecond time scales. Critical conditions such as queue occupancy change rapidly in timeframes measured in microseconds, whereas control traffic roundtrip latency may take in the order of milliseconds.

Resource reservation and flow admission control with in-band signaling are proposed in [10] to make sure an admitted flow would have an end-to-end latency within a maximum value. However, such guarantee is only eligible at the flow level. Latency-Based Forwarding (LBF) is a recent proposal that provides support for in-time and on-time services with precisely specified end-to-end latency objectives using a distributed path algorithm [11]. Each node on a path assesses whether the packet is on track for meeting its latency objective and determines a time budget for the packet to be forwarded. Local QoS actions and scheduling decisions take this budget into account, allowing packets to be slowed that can afford to while fast-tracking others as needed. LBF is close in spirit to the work presented here. Another proposal [12] builds on LBF and proposes an optimal scheduling algorithm that minimizes the average dwell time for all packets in the queue, but under the assumption that all packets are able to meet their deadlines under the scheduling algorithm. When this assumption cannot hold valid, the only option the router has is to drop the failing packets completely. Thus, relying merely on scheduling decisions to optimize dwell time is not good enough, the work presented in this paper allows to in addition make use of the possibility to reduce the time to

serialize and transmit packets in a queue by subjecting them to wash actions when needed.

The work that is presented here is unique in the fact that it proposes to leverage packet wash as a new tool, never used before, in the arsenal to achieve better QoS. As such, it does not represent a competing alternative for existing approaches but as a complementary approach that can be combined with other QoS approaches.

III. QUALITATIVE COMMUNICATION AND PACKET WASH

Qualitative Communication [13] was originally proposed to mitigate re-transmission overhead and associated delay when faced with slow or congested network conditions. Rather than having to drop a packet and its payload in its entirety as the only option when congestion occurs and queues are full, Qualitative Communication allows incremental portions of payload, referred to as *chunks*, to be dropped individually. Qualitative Communication provides senders with the ability to structure their payload into multiple chunks and assign them different priorities. This allows senders to differentiate between high-priority chunks that should not be dropped under any circumstance and lower priority chunks that might be acceptable to be dropped when it cannot be avoided. Chunk arrangement in the packet payload to show priority difference relies on the applications. Generally speaking, packet payload should be arranged such that the least significant chunks are at the tail of the packet payload. This way, dropping of a lower priority chunk becomes a truncation operation that can be more easily supported by hardware. Detailed encoding schemes (outside the scope of this paper) have been devised for various types of payload such as JPEG images, PNG images, and MPEG video data. In another variation [14][15], chunks can all be assigned equal priority, allowing any (but not all) chunks to be dropped up to a certain limit, for example in conjunction with random linear network coding schemes [16].

For any of this to happen, a new packet processing operation is introduced that is referred to as "Packet Wash". Packet Wash provides the functionality that reduces the size of a packet by selectively dropping chunks from the payload while ensuring that overall packet integrity is maintained. Packet headers are preserved and indication is given that the packet has in fact been washed as opposed to having been corrupted or compromised. Qualitative Communication and Packet Wash are included as one of the defining features of New IP [17], a novel networking protocol and framework that is aimed to address deficiencies of existing Internet Protocols and that has been evolved from BPP [18].

In some sense, Packet Wash exhibits certain commonalities with performing on-demand lossful compression: Ideally, packets are transferred in their entirety without any loss of payload or chunks, otherwise there would be no need to send them in the first place. However, under certain circumstances the controlled dropping of certain chunks as a last resort may be preferable over losing packet in their entirety, in particular when this means that extra delay due to the need for retransmission which otherwise would need to occur [19] can be avoided. Under those circumstances, that loss of chunks may result in slightly degraded quality of experience for applications that is still acceptable and preferrable over the alternative of requiring retransmission and imposing additional latency. This is similar to the case of lossful compression where, for example, bandwidth or storage space can be saved in exchange for slight degradations in image quality, for example reductions in resolution or color depth [20].

IV. A SYSTEM TO ACHIEVE HIGH-PRECISION LATENCY OBJECTIVES USING PACKET WASH

A. Overall Concept

The basic concept behind our design is that packets that are in danger of "being late" can potentially be accelerated if packet wash is applied to the packet, and possibly also to other packets ahead of it in the queue, in order to reduce the dwell time that the packet spends in the router. This acceleration can be applied irrespective of how the packets ended up in this particular order and queue in the first place, which means that the mechanism can applied in addition to and in conjunction with other QoS mechanisms, such as clever scheduling algorithms and prioritization schemes.

"Dwell time" is defined as the time that a packet spends in a network node before it is forwarded to the next hop. It is composed of three major components: (1) Processing delay: the time it takes the node to process the packet header, including tasks such as parsing and FIB lookup. (2) Queueing delay: the time that the packet spends in the egress queue before being transmitted. The queuing delay is affected by packet scheduling (determining the packet's placement into a queue) and the accumulated transmission delay of packets that are ahead in the queue. (3) Transmission delay: the time that it takes to serialize the packet and transmit it over the wire, generally proportional to the packet size.

In order to reduce dwell time, packet wash can be applied to the following effect:

1. It can be applied to the packet itself. As a result, the dwell time is shortened by the transmission delay that would otherwise be imposed by the chunk that is being dropped. The reduction in dwell time can be computed by the following formula: $D = S/B$, with D referring to the incurred transmission delay, S as the size of the dropped chunk, i.e. payload that no longer needs to be transmitted, and B as the link bandwidth. For example, dropping a chunk of 625 octets, i.e. 5000 bits, translates to a reduction in latency of 50 µsec in case of a 100 Mbps interface.
2. It can be applied to other packets that are ahead in the queue but that have not yet been transmitted. In that case, the dwell time can be reduced further proportional to the amount of payload savings that can be achieved across the packets in the queue. Expanding on the earlier example, in case of a queue depth of 100 packets with a wash potential of the same 625 octets (5000 bits) for just 20% of packets in the queue, savings of a full msec (20 packets * 50 µsec/packet) can be realized on a single hop. (It should be noted that beyond packet wash, lower priority packets ahead in the queue might also be dropped in their entirety.)

While applying packet wash and dropping chunks is acceptable as a last resort, these operations should not be applied lightly and only when truly necessary. After all, why would senders otherwise attempt to send these chunks in the first place. Therefore, a determination needs to be made when a packet's dwell time needs to be reduced. In general, this will be the case of an in-time service in which an end-to-end latency guarantee is given that must not be exceeded but which is at acute risk of being violated. To make this determination, the following processing needs to take place (Fig. 1), analogous to similar processing that occurs in conjunction with LBF [11]:

In the first step, the remaining Global Latency Budget (GLB), i.e. the time that is remaining to deliver the packet to its destination, is determined. One way to determine the GLB is to subtract the latency that the packet has incurred so far from the packet's end-to-end latency objective. With New IP, the objective can be carried along with the metadata for the latency measurement as part of the packet.

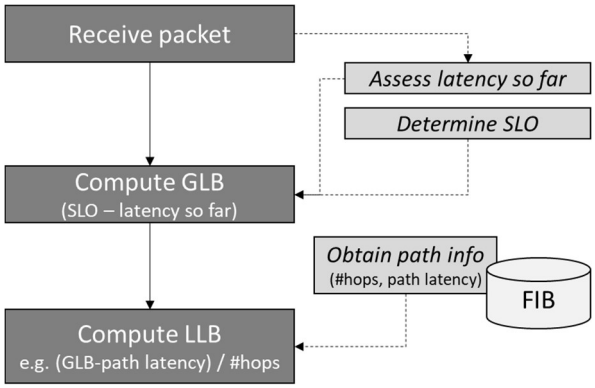

Fig. 1. Assessment of Local Latency Budget (LLB)

Next, the Local Latency Budget (LLB) that the packet can spend at this particular node needs to be determined. This determination needs to factor in the remainder of the path that still needs to be traversed, specifically the number of hops as well as the lower bound on the remaining path latency (i.e. the "non-negotiable" portion of the path latency that includes the propagation latencies, but not the more variable queueing latency). This data (remaining path latency, number of hops) can be attached to the destination's Forwarding Information Base (FIB) entry at the node. It can be provisioned using a controller or signaled to the node using IGP extensions that include e.g. link latencies.

Using this data, the LLB can now be computed. First, the remaining path latency is subtracted from the GLB, providing the variable portion of the latency across all the remaining nodes. From this, the local node's portion can be determined using a heuristic: It can be fully allocated to the local node. If the LLB is exhausted, it leaves no margin for nodes further downstream. It can be divided by the number of hops for a fairer allocation. Or it can be allocated in another manner, for example allocating the node a larger portion, taking the stance that extra actions to accelerate the packet should be avoided unless absolutely required while still leaving a small remaining margin for nodes further downstream.

The LLB can subsequently be used to schedule the packet accordingly. For example, Smallest Deadline First Scheduling (SDFS) [21] results in packets being scheduled in incremental order of their deadline, with the packets with the nearest deadline being scheduled first. In addition, to the point of this paper, when the LLB is so low that it would be missed once queuing delay is incurred, packet wash operations will be triggered. However, in cases where a packet is beyond being salvaged, for example when the LLB is already negative, no action would be taken or the packet might simply be discarded.

B. System design

The basic design of our system is depicted in Fig. 2. In essence, we are introducing two additional processing stages, one that is applied when the packet is enqueued, the other when the packet is dequeued for transmission. Both stages are hardware-friendly in their design and straightforward to add as part of a packet processing pipeline.

When a packet is enqueued, the Wash Trigger Assessor evaluates whether a packet is in danger of being late. This is done by comparing the local latency budget (determined as described earlier) and comparing it against the expected queueing and transmission delay (determined by the current queue depth and the interface bandwidth). If the LLB is less than the expected queuing and transmission delay, the packet is considered a "Wash Trigger Packet" (WTP) and the "wash now" state (*wns*) for the queue is activated.

More precisely, for a packet i that requires in-time guarantee, the input to the wash trigger assessor includes the packet per-hop deadline (LLB_i) and the packet dwell time in the router according to queue depth/occupancy (T_i). The Wash Trigger Assessor checks whether the condition $T_i \leq LLB_i$ can be satisfied. If that is the case (i.e. the dwell time remains below the permissible latency budget), no further action is needed and the packet is simply queued. If on the other hand the condition cannot be met, the Wash Trigger Assessor computes the amount of time that needs to be shaved off, or amount of payload that needs to be shaved off (named as Needed Truncation Amount, i.e. NTA). NTA_i is calculated as $(T_i - LLB_i) * B$, where B is the egress bandwidth. If NTA_i is an amount that could be met by washing the packets that are currently in the queue, the queue is put into "wash now" state.

As packets are dequeued for transmission, they are put through the second new processing stage, the Washer. The Washer applies wash operations to every washable packet as long as *wns* is active.

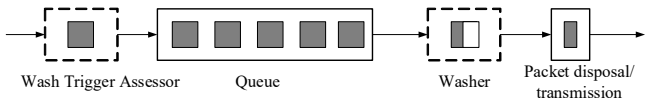

Fig. 2. Wash trigger assessor and washer

Once the WTP is processed by the washer and disposed off (transmitted or dropped), and no more WTP packets are in the queue, *wns* becomes inactive again. The state transition is shown in Fig. 3.

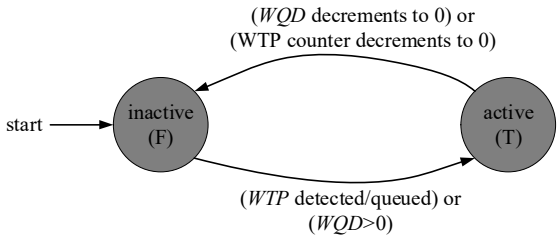

Fig. 3. *wns* finite state machine

Maintaining *wns* for a queue can occur in one of two ways:

- Option 1: For each queue, a "wash trigger packet gauge" is introduced that maintains the amount of current WTPs in the queue. Upon classification of a packet as WTP, the Wash Trigger Assessor marks the packet and increments the wash trigger packet gauge" by 1. Subsequently, the Washer decrements the gauge once the WTP-marked packet is dequeued.
- Option 2: For each queue, a "wash queue gauge" (WQD) maintains the queue depth for the most recent WTP. *wns* is active while $WQD > 0$. Whenever the Wash Trigger Assessor encounters a packet that is considered as a WTP, it sets WQD to the current queue depth. At the same time, the WQD is decremented for every packet that is dequeued while *wns* is active.

Option 2 is actually preferrable in that it is the easier to implement as it does not require any internal packet marking. It simply involves a simple assignment operation to re-set the WQD whenever a WTP is encountered, and a simple decrement operation whenever a packet is dequeued. It should also be emphasized that any state (WQD and *wns*) applies to the queue as a whole and is not specific to a flow or packet, minimizing the amount of state that needs to be retained.

The proposed scheme can be implemented efficiently in hardware per the following considerations: (1) when washable chunks reside at the end of the payload (as opposed to in arbitrary position), packet wash can be efficiently implemented as a (hardware-friendly) truncation operation. (2) keeping track of wash queue depth involves very simple arithmetic operations (increment, decrement) on a per-queue basis, conducted upon enqueue/dequeue operations, without needing to inspect contents inside the queue. The logic for this is extremely simple and does not involve multiple processing cycles. Although New IP is the context in which the presented concepts are being implemented, it could presumably be applied in conjunction with other protocols. As chunks are encoded as part of the payload, the presence of the scheme would be transparent to nodes that do not support it. In the worst case, things will revert to today's situation: retransmission will simply still be required and end-to-end latency objectives may be missed. As a result, the proposed scheme will not create any burden to the existing networks.

The proposed scheme does mean that one sender's traffic with very aggressive latency targets might potentially affect other senders' traffic whose less important chunks might be dropped. However, this will happen only within the legitimate terms of either sender's service, as the only packets that can be affected are those that are deemed washable, by definition concerning chunks that were designated as potentially disposable by the sender.

C. Examples

The principle of applying packet wash against a queue is depicted in Fig. 4. Packet 4, which is a WTP as well as washable, arrives at the tail of the outgoing queue. It is WTP marked. The number of washable packets in front of it including itself is 3 (Packet 1, 2, 4). Next, as shown in Fig. 5, the Packet 11 arrives at the tail of the outgoing queue, it is WTP marked. By now, packet 1 has already been transmitted out of the queue. The Packet 11 is not washable. Thus, the number of washable packets for the Packet 11 is 6 (Packet 2, 4, 6, 7, 8, 10).

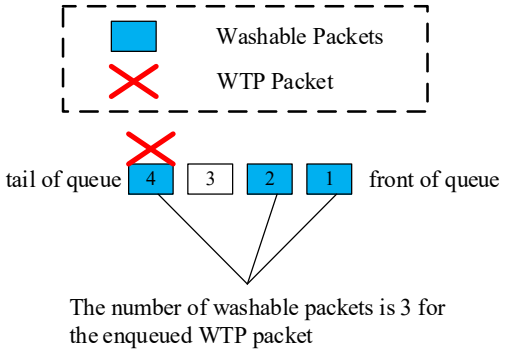

Fig. 4. Washable packets example 1

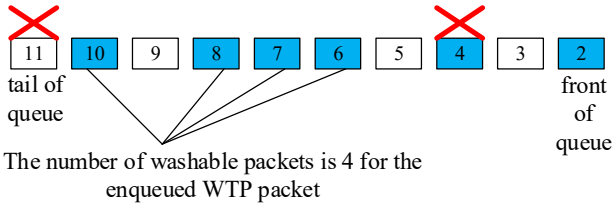

Fig. 5. Washable packets example 2

A more complex example to illustrate the operation of the system during runtime is depicted in TABLE I.

- "Packet arrival" indicates the arrival of new packet(s) needing to be enqueued.
- "Packet dequeue" indicates the dequeuing of packet(s) for transmission.
- "Packets in queue" contains all the packets that are currently in the queue and scheduled to be transmitted (newest arrivals on the left).
- "*wns*" indicates whether the wash now state is active or not.
- "# WTP" shows the number of WTPs in the queue. Each time a newly arriving packet is classified as WTP, the amount is incremented; when a WTP is dequeued, the amount is decremented.
- "WQD" maintains the queue depth for current WTPs, which is the number of packets in the queue from the beginning of the queue till the latest WTP.
- "Total AWA" (Total Allowable Wash Amount) maintains the total number of units (e.g. chunks of the same size, or multiples of some number of bytes) that are candidates for

washing currently in the queue. In this example, we assume that each washable packet will yield 5 units.

- "Total NTA" (Total Needed Truncation Amount) maintains the total number of units that need to be shaved off from the washable packets in order for the WTPs to satisfy the latency constraint in the current hop. Here, each WTP packet's NTA is set as 10 units.
- Washable packets are depicted in green/italics. WTPs are depicted in red/bold (and are washable for sake of this example).

TABLE I. PARAMTERS AND STATE MAINTAINED IN A ROUTER

Packet arrival	Packet deq.	Packets in queue	*wns*	# WTP	*WQD*	Total *AWA*	Total *NTA*
		*b*a	F	0	0	5	0
c		**c***b*a	T	1	3	10	10
	a	**c***b*	T	1	2	10	10
	b	**c**	T	1	1	5	5
	c	{}	F	0	0	0	0
d,e,f,g		g*fed*	F	0	0	15	0
h		**h**g*fed*	T	1	5	20	10
i,j	d	ij**h**g*fe*	T	1	4	15	5
k		**k**ij**h**g*fe*	T	2	7	20	15
l,m,n	e,f,g	nml**k**ij **h**	T	2	4	10	5
	h	nml**k**ij	T	1	3	5	5
..	…	…	…	…	…	…	

In the example, the arrival of the packet c triggers *wns*. WQD is set to 3. A single WTP is in the queue. As the packet a and b are dequeued, WQD is decremented accordingly. Once the packet c is dequeued, WQD is 0 and *wns* reverts to false. A few packet later, a WTP k arrives at a point when *wns* is already true (The packet h already triggers *wns* to be true). WQD is reset to the current depth of the queue (i.e. 7).

D. Avoiding Packet Overwash

In cases of a deep queue with many washable packets, there is a chance of subjecting more packets to wash operations and dropping more chunks than absolutely necessary to keep with the WTP's latency objectives. While dropping those chunks is by definition acceptable, doing so should only occur as a last resort in exceptional circumstances and still be avoided whenever possible (otherwise they would not need to be sent in the first place).

In order to avoid overwashing, the system can be refined to keep track of the wash potential currently in the queue ("AWA" in TABLE I.), as well as the savings that are achieved as packet wash operations are applied versus the amount determined to be needed, respectively the savings that still need to be achieved ("NTA" in TABLE I.). In that case, once the NTA for a queue drops to 0, wash operations cease and *wns* returns to inactive.

Further tweaks are conceivable, for example to ensure that WTPs are always subjected to washing before other packets, and to try to apply different measures of "fairness" as opposed to simply applying packet washes to the first packets in the queue as long as it is needed. However, it should be noted that such tweaks tend to add complexity combined with limited incremental benefits.

V. PERFORMANCE EVALUATION

In this section, we evaluate how the proposed packet wash mechanisms could improve the high precision end-to-end latency guarantee performance. Since packet wash is independently carried out at each intermediate router's egress point, as far as the per-hop deadline is met at each hop, the total transmission latency could be within the requested bounded latency and the packet is considered to be successful. Thus, we consider a simplified topology with n number of senders sending packets to m number of receivers. There is one intermediate router between the senders and receivers, and all packets towards the receivers require in-time guarantee and buffered in the same outgoing queue as shown in Fig. 6. On the other hand, every packet is washable and associated with a wash allowance ratio, which is defined as the ratio of packet payload size to the largest extend that is allowed to be truncated during the transmission.

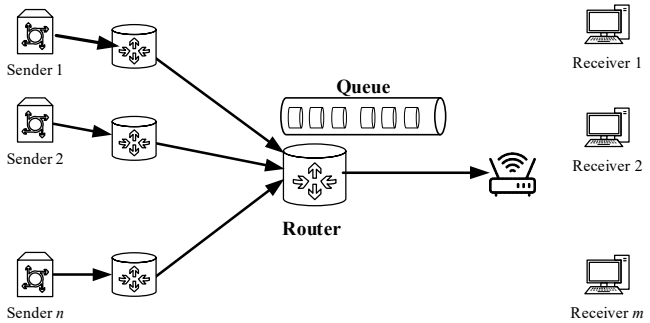

Fig. 6. Simplified topology

The following three scheduling schemes are included in the performance evaluation and comparisons:

- First In First Out (FIFO): which is the same as FCFS in [21]. The packets are scheduled according to their arrival time at the outgoing queue of the router. The packet which arrives earliest is scheduled first.
- Smallest Deadline First Schedule (SDFS) [21]: The packets are scheduled to be transmitted by the incremental order of deadline. The packet with the smallest deadline is scheduled first.
- Smallest allowance First Schedule (SAFS): The packets are scheduled to be transmitted by the incremental order of wash allowance size, which is calculated as the packet payload size multiplying with the wash allowance ratio.

Two types of performances are being evaluated: (1) packet delivery success ratio, which is defined as the ratio of the packets that meet their corresponding deadlines. The unsuccessful packets are dropped and cannot reach the receivers in time. (2) resulted packet wash ratio, which is defined as the ratio between the total truncated size and the total size of the packets' payload.

We build a simulator written in C++. The packets are generated randomly from a sender destinated to a receiver. The

packet size varies, the deadline of each packet is intentionally set up such that the packet would be dropped if the SDFS scheme is used, the wash allowance ratio may change in the simulations.

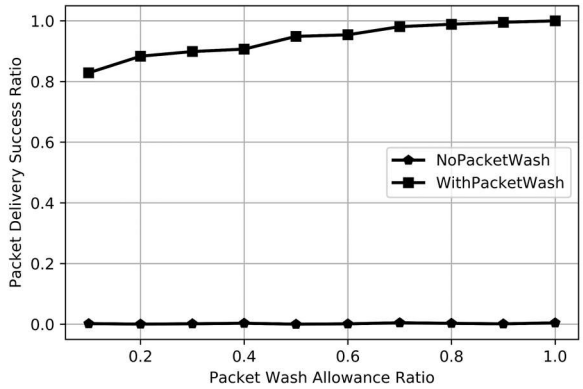

Fig. 7. Comparison of packet delivery success ratio when SDFS is adopted

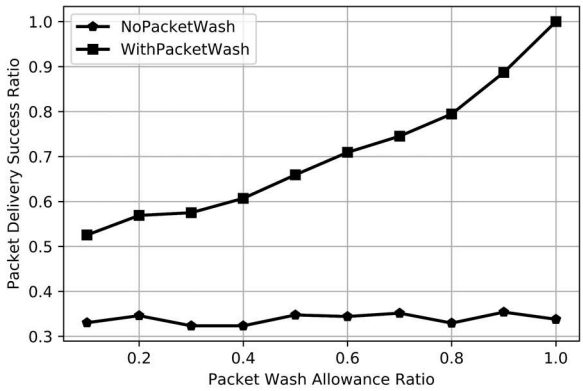

Fig. 8. Comparison of packet delivery success ratio when FIFO is adopted

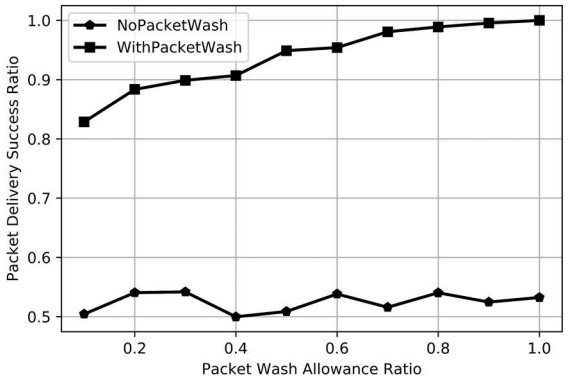

Fig. 9. Comparison of packet delivery success ratio when SAFS is adopted

Fig. 7, Fig. 8, and Fig. 9 show the comparison of packet delivery success ratio with and without packet wash, under the above three scheduling schemes respectively. Once the packet wash allowance ratio is set up, the simulation for NoPacketWash and WithPacketWash is re-run multiple times to get the average success ratio for both. (As a side note, Figures 8 and 9 do show slight variations in packet delivery success ratio for different packet wash allowance ratios even when packet was not even applied. This is explained by the fact that our simulation generates random packets for each run, causing the success ratio in each case to vary slightly.)

A packet's dwell time is reduced by applying packet wash to all packets in front of the packet and the packet itself. Since all the packets under SDFS without packet wash would fail, the packet delivery success ratio is 0, which has the worst performance. However, by applying packet wash to the packets in the queue, even with small wash allowance ratio (e.g. 10%), the packet delivery success ratio of SDFS with packet wash is dramatically increased to more than 80%. While the packet wash allowance ratio increases, more number of packets under the SDFS scheme can be delivered successfully to the receivers with their deadlines being met. When FIFO or SAFS is employed, originally without packet wash, the packet delivery success ratio is around 35% or 53% respectively (The packet delivery success ratio has small variance because the simulation is ran with randomness for packet generation, but averagely it does not differ much for FIFO or SAFS without packet wash). Unsurprisingly, packet wash also significantly increases the packet delivery success ratio when FIFO or SAFS is applied.

We can confidently conclude that the proposed packet wash mechanism can improve the high precision end-to-end latency guarantee performance under different packet scheduling algorithms.

On the other hand, Fig. 10 compares the packet delivery success ratio of FIFO, SDFS, and SAFS without packet wash, while Fig. 11 compares the packet delivery success ratio of FIFO, SDFS and SAFS with packet wash. In Fig. 11, the line of SDFS collides with SAFS, because they have the exactly same performance (this also applies to Fig. 12 for resulted packet wash ratio performance). The packet generation procedure results in the positioning of packets in the queue to be the same whether they are ordered incrementally by deadline or by wash allowance size.

Due to the packet generation procedure that is utilized in the simulations, SDFS without packet wash results in nearly zero packet delivery success ratio, in other words, all packets' dwell time exceed the corresponding configured deadline. FIFO and SAFS achieves better success ratio performance. When the proposed packet wash mechanism is applied, the packet delivery success ratio of SDFS and SAFS is increased to over 80%, which is especially exceptional for the scenario when SDFS is applied (from 0 to 80% success ratio). SDFS is largely adopted for scheduling flows in data centers, thus the proposed packet wash mechanisms is of great importance and effectiveness in such scenarios.

In addition, we observe from Fig. 12 that the resulted packet wash ratio is quite stable (around 30% to 40%) to achieve the high packet delivery success ratio for scenarios while SDFS or SAFS is adopted. When the packet wash allowance ratio is very small (e.g. 10%), the resulted packet wash ratio is small as well. It indicates that the packets' wash allowance is not fully used, because the wash allowance is not big enough to change a packet's delivery failure to success. When a packet is dropped

due to missing its deadline, packet wash would not be performed on the packets ahead in the queue.

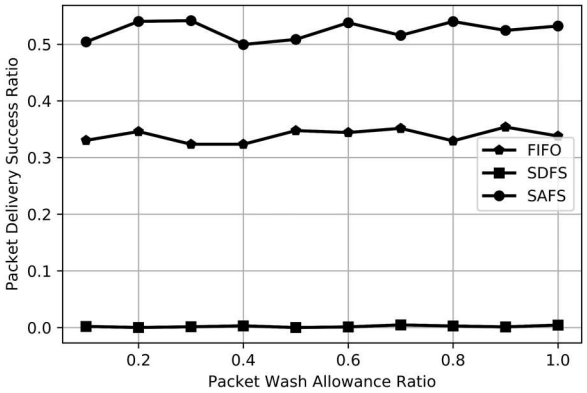

Fig. 10. Comparison of packet delivery success ratio under different scheduling schemes

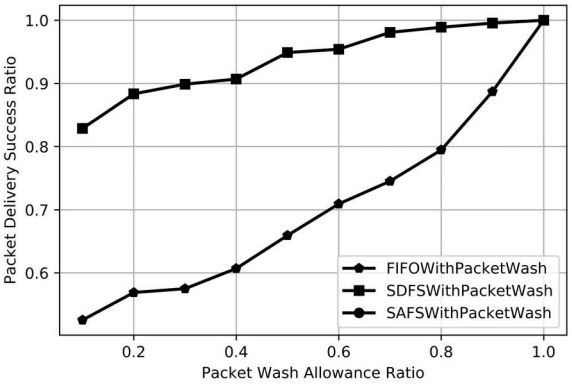

Fig. 11. Comparison of packet delivery success ratio with packet wash under different scheduling schemes

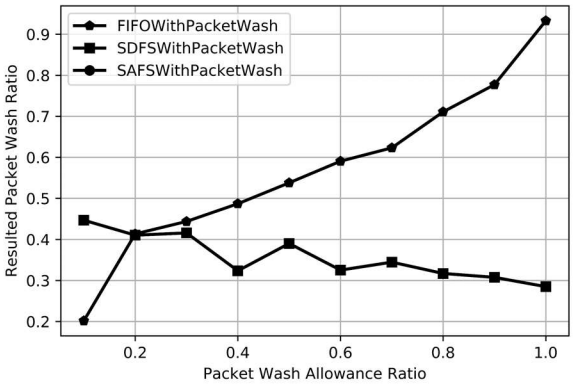

Fig. 12. Comparison of resulted packet wash ratio under different scheduling schemes

On the contrary, the resulted packet wash ratio for scenarios when FIFO is adopted increases constantly along with the packet wash allowance ratio. It indicates that in order to achieve higher packet delivery success ratio, FIFO does require packets ahead in the queue to be truncated more than SDFS and SAFS.

Taking both performances into consideration, we find out that SAFS is a scheduling policy that advances both FIFO and SDFS. SAFS utilizes reasonable portion of the packet wash allowance to achieve extremely high packet delivery success ratio.

VI. CONCLUSIONS

The existing QoS techniques built on the Best Effort principle of the current Internet can easily fail to comply with the stringent end-to-end service objectives. Qualitative Communications and the associated packet wash operation were introduced to reduce the number of packet drops when network congestion is encountered. This paper finds that the same mechanisms can also be useful to enhance high-precision services and facilitate end-to-end latency guarantees as demanded by many emerging applications. A packet in danger of being late can be accelerated by applying packet wash on itself as well as packets ahead in the queue when feasible, i.e. in conjunction with applications for which some controlled loss of lower-priority payload chunks is acceptable. To this end, two new processing stages are added to the egress interface that run a lightweight and hardware-friendly algorithm. Simulations confirm that proposed QoS technique improves the possibility of satisfying stringent end-to-end latency deadline under any scheduling schemes. Our proposed method is complementary to existing QoS approaches and allows for their seamless integration.

In the paper, we briefly discussed on how to avoid packet overwashing. For future works, the fairness for packet wash will be taken into further consideration. And the complexity of achieving fairness and the benefits provided to the packets need to be balanced and evaluated.

REFERENCES

[1] FG-NET-2030, Sub group 2, "New Services and Capabilities for Network 2030: Description, Technical Gap and Performance Target Analysis." https://www.itu.int/en/ITU-T/focusgroups/net2030/Documents/Deliverable/NET2030.pdf, 2019.

[2] A. Clemm, M.T.Vega, H.K. Ravuri, T. Wauters, F. De Turck, "Toward Truly Immersive Holographic-Type Communication: Challenges and Solutions". IEEE Communications Magazine Vol. 58 No. 1, January 2020.

[3] M. Wollschlaeger, T. Sauter, J. Jasperneite, "The Future of Industrial Communication: Automation Networks in the Era of the Internet of Things and Industry 4.0," IEEE Industrial Electronics Magazine Vol. 11 No. 1, March 2017.

[4] M. Maier, M. Chowdhury, B. P. Rimal, D. P. Van, "The Tactile Internet: Vision, Recent Progress, and Open Challenges," IEEE Communications Magazine Vol. 54 No. 5, May 2016.

[5] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, W. Weiss, "An Architecture for Differentiated Services," RFC 2475, IETF, December 1998.

[6] K. Ramakrishnan, S. Floyd, D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP," RFC 3168, IETF, September 2001.

[7] J. Wroclawski: "Specification of the Controlled-Load Network Element Service." RFC 2211, IETF, September 1997.

[8] S. Shenker, C. Partridge, R. Guerin: "Specification of Guaranteed Quality of Service." RFC 2212, IETF, September 1997.

[9] N. Finn, P. Thubert, B. Varga and J. Farkas: "Deterministic Networking Architecture." RFC 8655, IETF, October 2019.

[10] L. Han, Y. Qu, L. Dong, R. Li, "A Framework to Realize the Guaranteed Service for Bandwidth and Latency for Future IP network," 2020 Infocom workshop on New IP: The Next Step.

[11] A. Clemm, T. Eckert, “High-Precision Latency Forwarding over Packet - Programmable Networks,” IEEE/IFIP Network Operations and Management Symposium, 2020.

[12] L. Dong, R. Li, “Packet Level In-Time Guarantee: Algorithm and Theorems”, IEEE Globecom 2020.

[13] R. Li, K. Makhijani, H. Yousefi, C. Westphal, L. Dong, T. Wauters, and F. De Turck, “A Framework For Qualitative Communications Using Big Packet Protocol,” in NEAT’19: Proceedings Of The 2019 ACM Sigcomm Workshop On Networking For Emerging Applications And Technologies, pp. 22–28, ACM, 2019.

[14] L. Dong, R. Li, “In-Packet Network Coding for Effective Packet Wash and Packet Enrichment,” 2019 IEEE Globecom Workshop on Future Internet Architecture, Technologies and Services for 2030 and Beyond.

[15] L. Dong, K. Makhijani, R. Li, “Qualitative Communication Via Network Coding and New IP,” IEEE HPSR 2020.

[16] J. K. Sundararajan, D. Shah, M. Medard, S. Jakubczak, M. Mitzenmacher, J. Barros, “Network Coding Meets TCP: Theory and Implementation,” Proceedings of the IEEE Vol. 99 No. 3 pp. 490-512, 2011.

[17] R. Li, K. Makhijani and L. Dong, “New IP: A Data Packet Framework to Evolve the Internet,” IEEE HPSR 2020.

[18] R. Li, A. Clemm, U. Chunduri, L. Dong, and K. Makhijani, “A New Framework And Protocol For Future Networking Applications,” ACM Sigcomm Workshop on Networking for Emerging Applications and Technologies (NEAT 2018), pp. 637–648, May 2018.

[19] N. Cardwell, S. Savage, T. Anderson, “Modeling TCP Latency,” IEEE INFOCOM 2000.

[20] N. Ponomarkenko, S. Krivenko, V. Lukin, K. Dglazarian, J. Astola, “Lossy Compression of Noisy Images Based on Visual Quality: A Comprehensive Study,” EURASIP Journal on Advances in Signal Processing, Article ID 976436, 2010.

[21] C. Wilson, H. Ballani, T. Karagiannis, and A. Rowtron, “Better never than late: Meeting deadlines in datacenter networks,” SIGCOMM Computer Communication Review, vol. 41, no. 4, p. 50, 2011.

[22] V. K. Gopalakrishna, Y. Kaymak, C. Lin and R. Rojas-Cessa, "PEQ: Scheduling Time-Sensitive Data-Center Flows using Weighted Flow Sizes and Deadlines," 2020 IEEE 21st International Conference on High Performance Switching and Routing (HPSR).